DIAGONAL TESTING METHOD FOR FLASH MEMORIES

BACKGROUND OF THE INVENTION

1. Field of the Invention

The present invention relates to a testing method for flash memories, particularly to a diagonal testing method for flash memories.

2. Background of the Invention

The flash memory is a type of programmable and erasable non-volatile memory for recording data. Recently, the requirement for flash memories has been growing rapidly, especially in the application of wireless transmission, automatic equipment, data storage, set-top box and all kinds of multi-media products. The flash memory also has been largely integrated into the SOC (system on a chip) comprising all kinds of logic circuits and memory cores, particularly in the portable product with battery as its power supply. Because the design for SOC tends to have high density, high capacity and high pins, the test time and difficulties are significantly increased. Thus, it becomes an important issue to rapidly and effectively reduce the test time to increase the yield.

Generally, the testing or diagnosis for flash memories is more difficult than conventional memories, particularly due to the disturbance problem during reading and writing cells on the same row or same column. It regulates the disturbance problem and the influence for flash memories in the IEEE 1005 standard. For example, the adjacent memory cells will have undesired charge transfer during the reading or programming of the memory cell in high voltage operation. Compared with the operation of SRAM or DRAM, the flash memory is different in not only writing a single bit or word, but on the contrary, that it can proceed the flash erase on the entire block or chip, such as writing the entire block or chip with logic 1 simultaneously.

Recently, there are different testing methods developed continuously. In which, disturb faults and an optimum algorithm (hereinafter referred to "EF") are disclosed in M. G. Mohammad, K. K. Saluja, and A. Yap, "Testing Flash Memories," in Proc. 13th Int. Conf. VLSI Design, Jan. 2000, pp. 406-411. In a paper proposed by M. G. Mohammad and K. K. Saluja, "Flash Memory Disturbances: Modeling and Test," Proc. IEEE VLSI Test Symp., Marina Del Ray, California, April 2001, pp. 218-224, it divides the disturb fault model for flash memories into several coupling faults, and proposes an effective March-like algorithm (hereinafter referred to Flash March) for detecting the coupling faults. In a paper proposed by J. C. Yen, C. F. Wu, K. L. Cheng, Y. F. Chou, C. T. Huang, and C. W. Wu, "Flash Memory Built-in Self-test Using March-Like Algorithms", Proc. IEEE Int. Workshop on Electronic Design, Test, and Applications (DELTA), Christchurch, Jan. 2002, pp. 137-141, a bit-oriented and word-oriented March-like algorithm (hereinafter referred to March-FT) is disclosed, and is used to cover all the disturb fault types defined in IEEE 1005 standard.

For the operation of the flash memory, the operational speed for erasing is the slowest, the speed for programming is faster, and the speed for reading is the fastest. However, the conventional March-like algorithm includes many writing operations (including erasing and programming), and the test time is too long to make the testing cost relatively high. Therefore, how to design an effective testing method for effectively reducing the required test time without reducing the fault coverage becomes a very important issue.

SUMMARY OF THE INVENTION

The main object of the present invention is to provide a diagonal testing method for flash memories that can effectively reduce the required test time without reducing the fault coverage.

The second object of the present invention is to provide a testing

method for flash memories that cannot only be applied in the production of testing software, but also be applied in the production of built-in self-testing circuit.

To this end, the diagonal testing method for flash memories according to the present invention includes steps (a) to (e). In step (a), the entire flash memory cell array is erased. In step (b), the cells except the first diagonal are programmed. In step (c), the cells in the first diagonal are read. In step (d), the cells in the first diagonal are programmed. In step (e), the cells except the first diagonal are read.

The diagonal testing method for flash memories according to the present invention further includes steps (f) and (g). In step (f), the cells in the second diagonal are programmed. In step (g), the cells in the first diagonal are read. Steps (f) and (g) and steps (a) to (e) need not have the sequential relations during executing. The developer for the testing program can decide whether steps (f) and (g) or steps (a) to (e) are executed first according to the convenience for developing program.

The diagonal testing method for flash memories according to the present invention further includes steps (h) to (j). In step (h), the flash memory cell array is erased. In step (i), the cells in the first diagonal of the flash memory cell array are programmed in the direction opposite to the first diagonal. In step (j), the cells in the flash memory cell array except the first diagonal are read. Steps (h) to (j), steps (a) to (e) or steps (f) and (g) need not have the sequential relations during executing. The developer for the testing program can decide whether steps (h) to (j), steps (f) and (g), or steps (a) to (e) are executed first according to the convenience for developing program.

In step (j) according to the present invention, an extended algorithm can be used to make the fault coverage up to 100%. But, when the extended algorithm is used, step (f) can be omitted.

In the above-mentioned method, the flash memory can be regarded as several squares, and executed in the direction from top to down and from left to right. Each square is provided with a first diagonal in -45 degrees from the upper left to the lower right, and a second diagonal in +45 degrees from the lower left to the upper right. The present invention is to program the cells in the first diagonal or the second diagonal, and then read the cells except the first diagonal or the second diagonal; or, program the cells except the first diagonal or the second diagonal, and then read the cells in the first diagonal or the second diagonal so as to detect the disturb fault in the flash memories and normal memory fault models.

The above-mentioned -45 and +45 degrees in the execution direction for the first diagonal and the second diagonal respectively are only one embodiment of the present invention. The designer can modify the degrees based on his own preference in the practical application, and the execution direction of the present invention is not limited to a specific direction, such as the first diagonal or the second diagonal.

The fault detecting range in the present invention covers not only the disturb faults in the flash memories, but also the conventional memory faults.

The circuit area formed by the above-mentioned testing method is relatively small, which is about 2,000 to 3,000 gate counts for a common flash memory. The testing method according to the present invention can reduce the test time to about 42.69% as the conventional March-like algorithm. Thus, the testing method according to the present invention can effectively reduce the required test time without reducing the fault coverage, which obviously presents the innovation and the progress.

BRIEF DESCRIPTION OF THE DRAWINGS

The present invention will be described according to the appended drawings in which:

FIGS. 1(a) to 1(c) are hint diagrams of the diagonal divisions of the flash memory according to the present invention;

FIG. 2 is a testing flow chart of the first embodiment according to the present invention;

FIGS. 3(a) to 3(g) are exploded diagrams of the first embodiment according to the present invention;

FIGS. 4(a) to 4(i) are other exploded diagrams of the first embodiment according to the present invention;

FIGS. 5(a) to 5(i) are exploded diagrams of the second embodiment according to the present invention;

FIGS. 6(a) to 6(i) are exploded diagrams of the third embodiment according to the present invention;

FIGS. 7(a) to 7(i) are exploded diagrams of the fourth embodiment according to the present invention;

FIGS. 8(a) to 8(i) are exploded diagrams of the fifth embodiment according to the present invention;

FIGS. 9(a) to 9(h) are exploded diagrams of the sixth embodiment according to the present invention;

FIGS. 10(a) to 10(h) are exploded diagrams of the seventh embodiment according to the present invention; and

FIGS. 11(a) to 11(h) are exploded diagrams of the eighth embodiment according to the present invention.

DESCRIPTION OF THE PREFERRED EMBODIMENTS

The diagonal testing method for flash memories according to the present invention is mainly to eliminate the disturb faults in the flash

memory and the fault models appearing in the common memory. Because a high voltage is applied to the terminals of the floating gate of the flash memory, the disturbance problem is likely to happen, and the cells influenced by the disturbance effect are usually located on the same bit-line or the same word-line as the programmed or read cells. Generally, the disturb fault model can be divided into:

1. Word-line Program Disturbance (WPD). The reason for this disturbance is that after programming with a cell, the unprogrammed cells on the same word-line are also programmed.

2. Word-line Erase Disturbance (WED). The reason for this disturbance is that after programming with a cell, the programmed cells on the same word-line are erased.

3. Bit-line Program Disturbance (BPD). The reason for this disturbance is that after programming with a cell, the unprogrammed cells on the same bit-line are also programmed.

4. Bit-line Erase Disturbance (BED). The reason for this disturbance is that after programming with a cell, the programmed cells on the same bit-line are erased.

5. Read Disturbance (RD). The reason for this disturbance is that after repeatedly reading a cell, the current leakage for the floating gate cannot be retained at the status of logic zero.

6. Over Erase (OE). The reason for this disturbance is that if a cell is repeatedly erased, such as after erasing, the threshold voltage is too low to convert the cell into a depletion transistor.

A normal fault model for the memory includes the followings:

1. Stuck-At Fault (SAF)

2. Transition Fault (TF)

3. Stuck-Open Fault (SOF)

4. Address decoder Fault (AF)

5. State-Coupling Fault (SFst)

The algorithm according to the present invention is based on the reading and writing operations for the two diagonals in the flash memory cell array. The first diagonal, as the solid lines in the FIGS. 1(a) to 1(c), is a -45° line from the upper left to the lower right. The second diagonal, as the dashed lines in FIGS. 1(a) to 1(c), is a +45° line from the lower left to the upper right, wherein m represents the number of rows, and n represents the number of columns, and the lowest address is located at the upper left. The algorithm according to the present invention can be applied to the flash memory cell array with different numbers of rows and column. For example, FIG. 1 (a) represents the case that the number of rows is equal to the number of columns in the flash memory cell array, FIG. 1 (b) represents the case that the number of rows is less than the number of columns, and FIG. 1 (c) represents the case that the number of rows is larger than the number of columns. As shown in FIG. 1(b) and FIG. 1(c), the testing method is to assume that the flash memory is composed of cells of several squares, and executed in the direction from top to bottom and from left to right. For convenient description, the following embodiments having the flash memory cell array with the number of rows less than and equal to the number of columns are described.

FIG. 2 is a testing flow chart of the first embodiment according to the present invention, which includes the following steps, and the diagrams for each step are shown from FIGS. 3(a) to 3(g):

STEP 21: Erasing the entire flash memory cell array, so that the values of entire flash memory cell array are set as logic one.

STEP 22: As shown in FIG. 3(a), which can be divided into three sub-steps:

I. Reading the cells except the first diagonal. If the values thereof are not logic one, it means the SOF and CFst faults exist.

II. Programming the cells except the first diagonal, for detecting the cells in the first diagonal if occurred with WPD and BPD faults.

III. Reading the cells except the first diagonal. If the values thereof are not logic zero, it means the OE fault exists.

STEP 23: As shown in FIG. 3(b), which can be divided into three sub-steps:

I. Reading the cells in the first diagonal. If the values thereof are not logic one, it means the WPD and BPD faults exist.

II. Programming the cells in the first diagonal, for detecting the cells except the first diagonal if occurred with the WED and BED faults.

III. Reading the cells in the first diagonal. If the values thereof are not logic zero, it means the OE fault exists.

STEP 24: As shown in FIG. 3(c), reading the cells except the first diagonal. If the values thereof are not logic zero, it means the WED and BED faults exist.

STEP 25: As shown in FIG. 3(d), programming the cells in the second diagonal, for detecting the cells in the first diagonal if occurred with the WED and BED faults.

STEP 26: As shown in FIG. 3(e), reading the cells in the first diagonal. If the values are not logic zero, it means the WED and BED faults exist.

STEP 27: Erasing the entire flash memory cell array, for setting the entire flash memory cell array as logic one.

STEP 28: As shown in FIG. 3(f), which can be divided into three sub-steps:

I. Reading the cells in the first diagonal in the direction opposite to the first diagonal for detecting more common memory faults. If the values thereof are not logic one, it means the SOF and CFst faults exist.

II. Programming the cells in the first diagonal in the direction opposite to the first diagonal for detecting the cells except the first diagonal if occurred with the WPD and BPD faults.

III. Reading the cells in the first diagonal in the direction opposite to the first diagonal for detecting more common memory faults. If the values thereof are not logic zero, it means the CFst faults exist.

STEP 29: As shown in FIG. 3(g), reading the cells except the first diagonal. If the values thereof are not logic one, it means the WPD and BPD faults exist.

One embodiment of the algorithm according to the testing method of the present invention is shown as follow:

1. Flash Erase;

// let *i* represents the row address index of the flash memory cell array, *j* represents the column address index of the flash memory cell array, and *x* represents the minimum between *m* and *n*

2. for *j*=0 to (*n*-1) // except the cells in the first diagonal

for *i*=0 to (*m*-1)

if (((*j*-*i*) % *m*)!=0)

{ Read1 (*i*,*j*);

```
                Program (i,j);
                Read0 (i,j);}
3.  for j=0 to (n-1)        //  the cells in the first diagonal
      for i=0 to (m-1)
5       if (((j-i) % m)=0)
            {Read1 (i,j);
                Program (i,j);
                Read0 (i,j);}
4.  for j=0 to (n-1)        //  except the cells in the first diagonal
10      for i=0 to (m-1)
          if (((j-i) % m)!=0)        Read0 (i,j);
5.  for j=0 to (n-1)        //  the cells in the second diagonal
      for i=(m-1) to 0
          if (((j-((m-1)-i)) % m)=0)   Program (i,j);
15  6.  for j=0 to (n-1)        //  the cells in the first diagonal
      for i=0 to (m-1)
          if (((j-i) % m)=0)        Read0 (i,j);
7.  Flash Erase;
8.  for j=(n-1) to 0    //  the cells in the first diagonal
20      for i=(m-1) to 0
      if   (((j-i) % m)=0)
          {   Read1 (i,j);
              Program (i,j);
              Read0 (i,j);        }
```

9. for j=0 to (n-1) // except the cells in the first diagonal

for i=0 to (m-1)

if (((j-i) % m)!=0) Read1 (i,j);

The algorithm of the present invention can be represented as the following simple form:

$$\{(E);\underset{!D_1}{\Uparrow}(R1,P,R0);\underset{D_1}{\Uparrow}(R1,P,R0);\underset{!D_1}{\Uparrow}(R0);\underset{D_2}{\Downarrow}(P);\underset{D_1}{\Uparrow}(R0);(E);\underset{D_1}{\Downarrow}(R1,P,R0);\underset{!D_1}{\Uparrow}(R1)\}$$

wherein the symbol $\Uparrow$ represents an ascending address sequence, the symbol $\Downarrow$ represents a descending address sequence, D_1 represents the cells in the first diagonal, D_2 represents the cells in the second diagonal, $!D_1$ represents except the cells in the first diagonal, $!D_2$ represents except the cells in the second diagonal, $R1$ represents to read cells, and if the value is not logic one, an error will be detected; P represents to program cells; $R0$ represents to read cells, and if the value is not logic zero, an error will be detected.

It should be noted that, the fault model and the algorithm in the above-mentioned embodiments are illustrated with the NOR gate type of flash memories, but there is no specific limit in the practical application. Moreover, for the RD fault, the operating voltages for reading a cell are equal to those of the effected cells of the WED fault, and therefore, the RD fault will be activated and detected as a WED fault.

The algorithm for the above-mentioned embodiment according to the present invention cannot reach 100% for the fault coverage on the AF and CFst faults. An available method as follows can be used to modify the step 29:

STEP 29: As shown in FIG. 3(b), which can be divided into three sub-steps:

I. Reading the cells except the first diagonal in the direction from

the higher address to the lower address. If the values thereof are not logic one, it means the SOF and CFst faults exist.

II. Programming the cells except the first diagonal in the direction from the higher address to the lower address.

III. Reading the cells except the first diagonal in the direction from the higher address to the lower address. If the values thereof are not logic zero, it means the OE fault exists.

An embodiment of the algorithm according to amended step 29 is as follows:

10. for j=(n-1) to 0 // (*R1,P,R0*) all cells
for i=(m-1) to 0 // except the cells in the first diagonal
if (((j-i) % m)!=0)
{ Read1 (i,j);
Program (i,j);
Read0 (i,j);}

The algorithm of the present invention can be represented as the following simple form:

$$\{(E); \Uparrow_{!D_1}(R1,P,R0); \Uparrow_{D_1}(R1,P,R0); \Uparrow_{!D_1}(R0); \Downarrow_{D_2}(P); \Uparrow_{D_1}(R0); (E); \Downarrow_{D_1}(R1,P,R0); \Downarrow_{!D_1}(R1,P,R0)\}$$

The test time for the modified algorithm (hereinafter referred to an extended diagonal type) is slightly increased, and the testing length becomes $2e + (2mn + max(m, n))p + 5mnr$, where the parameter e represents the erasing time, the parameter p represents the programming time, and the parameter r represents the reading time. For a 2MB flash memory, the test time is about 6.711 seconds, which is almost equal to the March-like algorithm. However, if it only considers the fault detection of the disturb fault model in the flash memory, the algorithm of the first

embodiment according to the present invention is preferred. Because it is provided with shorter test time, the testing cost can be greatly reduced.

FIGS. 4(a) to 4(i) are other exploded diagrams of the first embodiment (hereinafter referred to the diagonal type) according to the present invention. It should be noted that, the steps in FIGS. 4(g) to 4(i) are more independent flows relative to the steps in FIGS. 4(a) to 4(f), so that the steps in FIGS. 4(g) to 4(i) can be configured before the steps in FIGS. 4(a) to 4(f). The sorted flows are shown as FIGS. 5(a) to 5(i), and are the second embodiment according to the present invention.

FIGS. 6(a) to 6(i) are the exploded diagrams for the testing steps of the third embodiment according to the present invention, which are improved step procedures of the first embodiment in FIGS. 4(a) to 4(i). The third embodiment is to move the steps in FIGS. 4(e) and 4(f) to the end of the entire process (hereinafter referred to a modified diagonal type).

It should be noted that, the steps in FIGS. 6(e) to 6(i) are independent flows relative to the steps in FIGS. 6(a) to 6(d), so that the steps in FIGS. 6(e) to 6(i) are configured before the steps in FIGS. 6(a) to 6(d). The sorted flows are shown as FIGS. 7(a) to 7(i), and are the fourth embodiment according to the present invention.

FIGS. 8(a) to 8(i) are the exploded diagrams for the testing steps of the fifth embodiment according to the present invention, which uses the algorithm of the extended diagonal type (hereinafter referred to an extended diagonal type), so as to achieve 100% fault coverage.

FIGS. 9(a) to 9(h) are the exploded diagrams for the testing steps of the sixth embodiment according to the present invention, which improves the step procedure for the fifth embodiment in FIGS. 8(a) to 8(i). The sixth embodiment is to move the step of FIG. 8(f) as the last step in the testing flow, and omit the step of FIG. 8(e) (hereinafter referred to an extended and modified diagonal type).

Similar to the above cases, the steps in FIGS. 9(e) to 9(h) are independent flows relative to the steps in FIGS. 9(a) to 9(d), so that the steps from the FIGS. 9(e) to 9(h) are configured before the steps in FIGS. 9(a) to 9(d). The sorted flows are shown as FIGS. 10(a) to 10(h) for the seventh embodiment according to the present invention.

FIGS. 11(a) to 11(h) are exploded diagrams for the testing steps of the eighth embodiment according to the present invention, which improves the step procedures of the fifth embodiment in FIGS. 8(a) to 8(i). The eighth embodiment is to move the steps 8(g) to 8(i) to the start of the testing flow.

Table 1 illustrates the characteristics of test time and testing length comparing the present invention with the prior art through the RAMSES-FT fault detection simulator (referred to the paper by K. L. Cheng, J. C. Yeh, C. W. Wang, C. T. Huang, and C. W. Wu, "RAMSES-FT: A Fault Simulator for Flash Memory Testing and Diagnostics", Proc. IEEE VLSI Test Symp., Monterey, California, Apr. 2002). Table 2 illustrates the comparison result for the fault coverage between the present invention and the prior art. The above-mentioned tests makes the example of the 2MB (256K x 8) flash memory, where the erasing time is 200ms, the bit programming time is 12μs, and the bit reading time is 10ns. Furthermore, the above-mentioned test only uses the bit-oriented algorithm.

Table 1

algorithm	Test Length	Complexity ($N=m \times n$)			Test time (@100MHz ; unit: sec)
		erase	program	read	
EF	$2e+(mn+2m+n-2)p+(2mn+m+n-3)r$	2	$1N+3\sqrt{N}$	$2N+2\sqrt{N}$	3.569
Flash March	$2e+(2mn)p+(4mn)r$	2	$2N$	$4N$	6.702
March-FT	$2e+(2mn)p+(6mn)r$	2	$2N$	$6N$	6.707

diagonal type	$2e+(mn+2max(m,n))p+(4mn+max(m,n))r$	2	$1N+2\sqrt{N}$	$4N+\sqrt{N}$	3.569
modified diagonal type	$2e+(mn+2max(m,n))p+(4mn+3max(m,n))r$	2	$1N+2\sqrt{N}$	$4N+3\sqrt{N}$	3.569
extended diagonal type	$2e+(2mn+max(m,n))p+(5mn)r$	2	$2N+\sqrt{N}$	$5N$	6.711
extended and modified diagonal type	$2e+(2mn)p+(5mn)r$	2	$2N$	$5N$	6.707

Table 2

	WPD	WED	BPD	BED	OE	RD
EF	100%	100%	0%	100%	100%	100%
Flash March	100%	100%	100%	100%	100%	100%
March-FT	100%	100%	100%	100%	100%	100%
diagonal type	100%	100%	100%	100%	100%	100%
modified diagonal type	100%	100%	100%	100%	100%	100%
extended diagonal type	100%	100%	100%	100%	100%	100%
extended and modified diagonal type	100%	100%	100%	100%	100%	100%

Table 2 (continue)

	SAF	TF	SOF	AF	CFst	Total fault coverage
EF	100%	87.5%	12.5%	44.5%	50%	72.23%

Flash March	100%	100%	50%	100%	75%	93.18%
March-FT	100%	100%	100%	100%	100%	100%
diagonal type	100%	100%	100%	72.5%	86.25%	96.25%
modified diagonal type	100%	100%	100%	81.6%	89.15%	97.34%
extended diagonal type	100%	100%	100%	100%	100%	100%
extended and modified diagonal type	100%	100%	100%	100%	100%	100%

In view of the results from Table 1 and Table 2, it has been proved that the testing length in the present invention is shorter, and is provided with better fault coverage for the disturb faults and common memory faults. Moreover, the testing method according to the present invention is provided with the regularity and symmetry, and makes the built-in self-test circuit built with the diagonal testing method according to the present invention easier to produce.

The above-described embodiments of the present invention are intended to be illustrative only. Numerous alternative embodiments may be devised by those skilled in the art without departing from the scope of the following claims.